Web Application Security

Kitisak Jirawannakool

Electronics Government Agency (Public Organization)

Agenda

- What is Security?
- Web Application Security
- Real cases
- Securing CMS tactics
- Web Application Security Testing

What is Security?

- C (Confidentiality)
- I (Integrity)
- A (Availability)

CIA
Security

Security Myths : We are not a target

- "Mostly I hear it from victims. They think they aren't worth hacking. Some say it's not worthwhile because they're a small business – not on anybody's radar. Others contend they don't collect Social Security numbers, credit card data or other 'valuable' information. They are usually wrong."

Alan Brill, senior managing director for the cybersecurity and information assurance practice at Kroll

What is Secure Software?

It’s secure! Looks at the lock, up on the left!

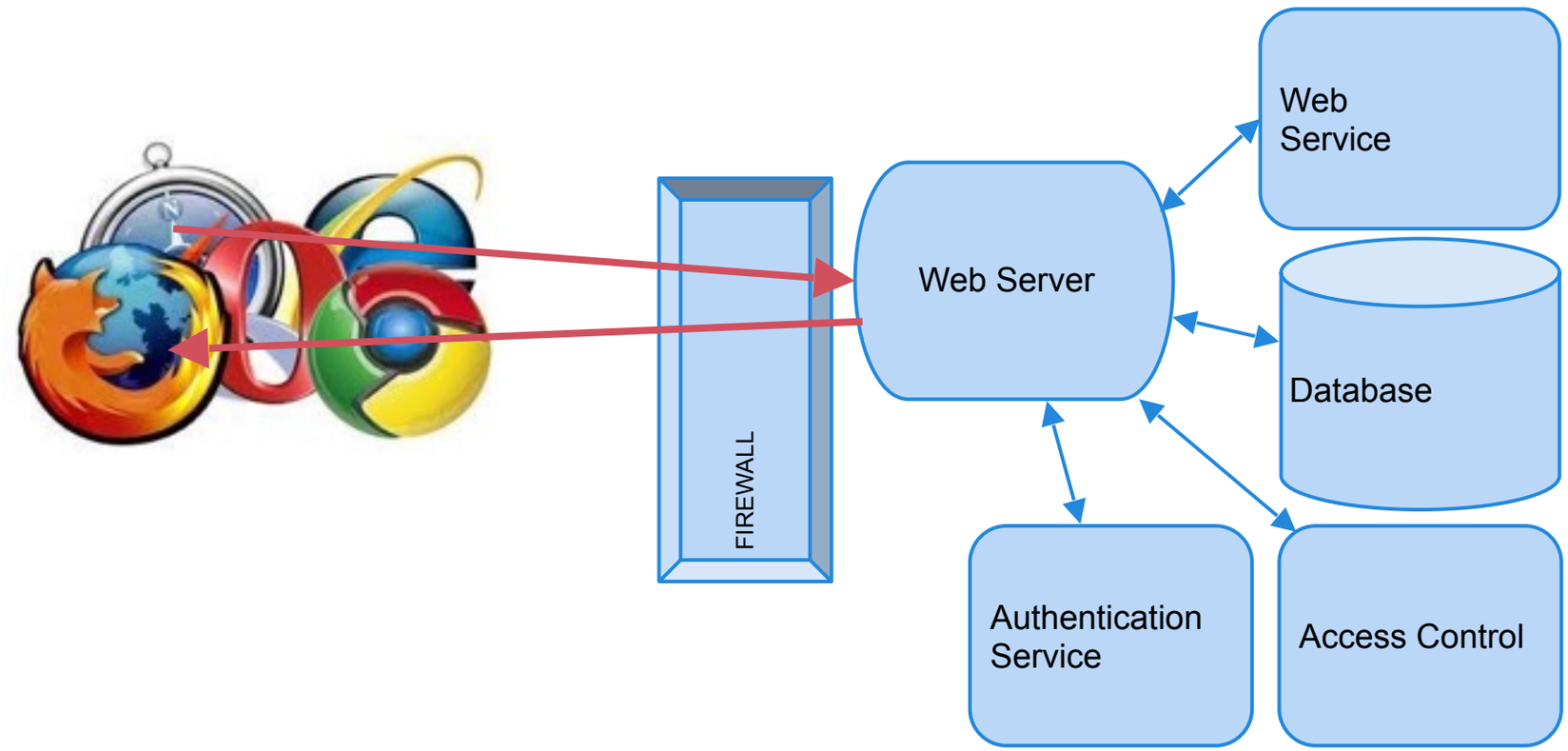
Sure! The news said that is unbreakable!

It’s secure! It’s Google!

Software Security Principles

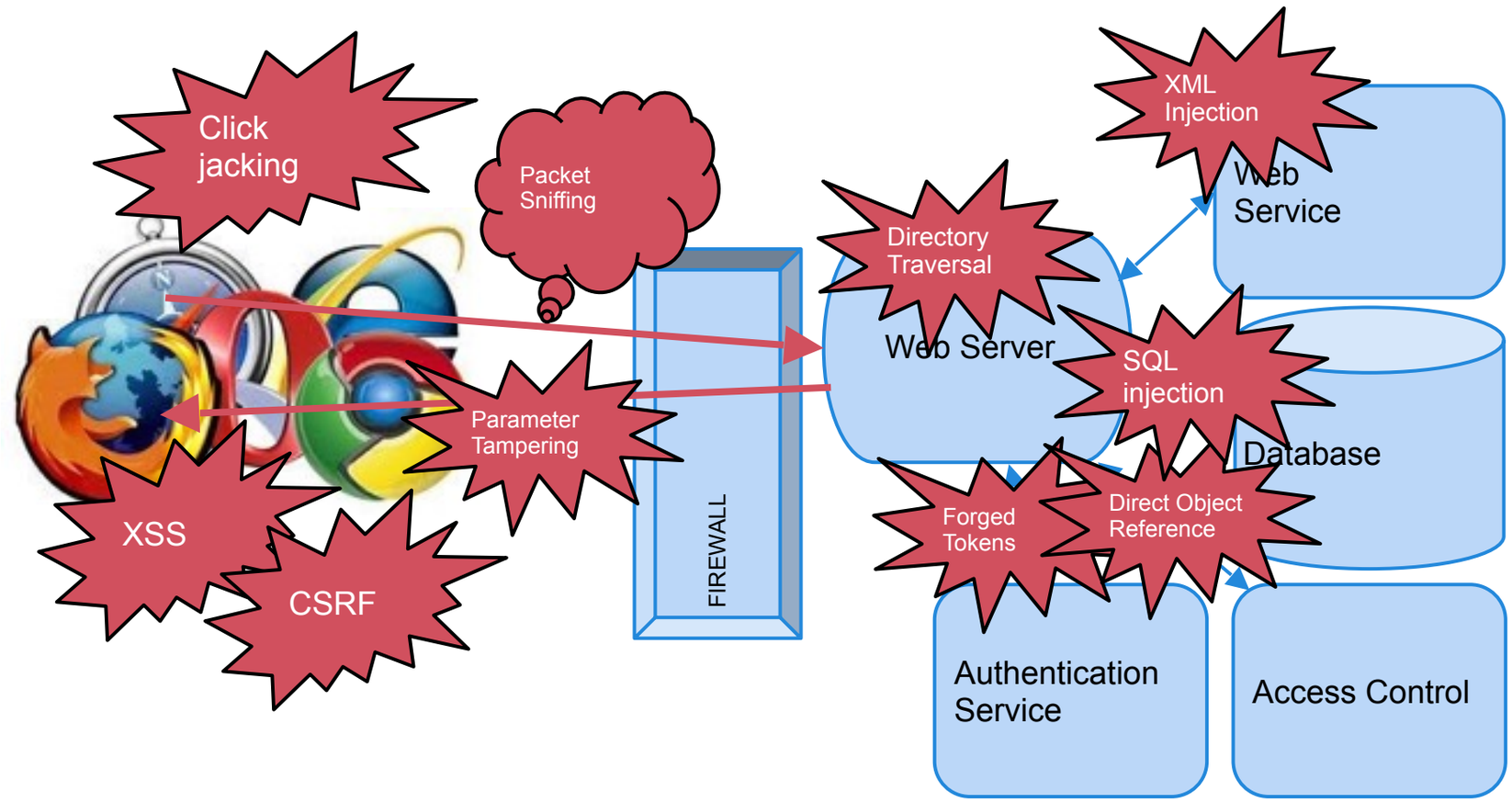
- Security vulnerabilities in the software development process are expected.
- The control of the security bugs and flaws in the software should be considered as part of the process of software development.
- Vulnerability management (fixing process) is the most important step of the process of software security.

Web Architecture Components



Reference :: Web Application Hacking/Security 101
(https://docs.google.com/presentation/d/1fw7fO7kmVTcfXuupGTezSM76cdQH3IbYos5xu95LyMs/edit#slide=id.p)

Web Architecture Attacks



Reference :: Web Application Hacking/Security 101
(https://docs.google.com/presentation/d/1fw7fO7kmVTcfXuupGTezSM76cdQH3IbYos5xu95LyMs/edit#slide=id.p)

Hack for ?

- Defacing website
  - http://www.zone-h.org
- Phishing
- Spreading malware
- Making money
- Discrediting opposite
- Fun
- State sponsor

Hot topic!!!!!!

South Korean banks and media report computer network crash, causing speculation of North Korea cyberattack

Published March 20, 2013 / Associated Press

RECOMMENDED

Sex-enhancing sho

Dr. Ben Carson ste
the show at CPAC

TRENDING IN WO

1 South K
media re
network
speculat
cyberatt

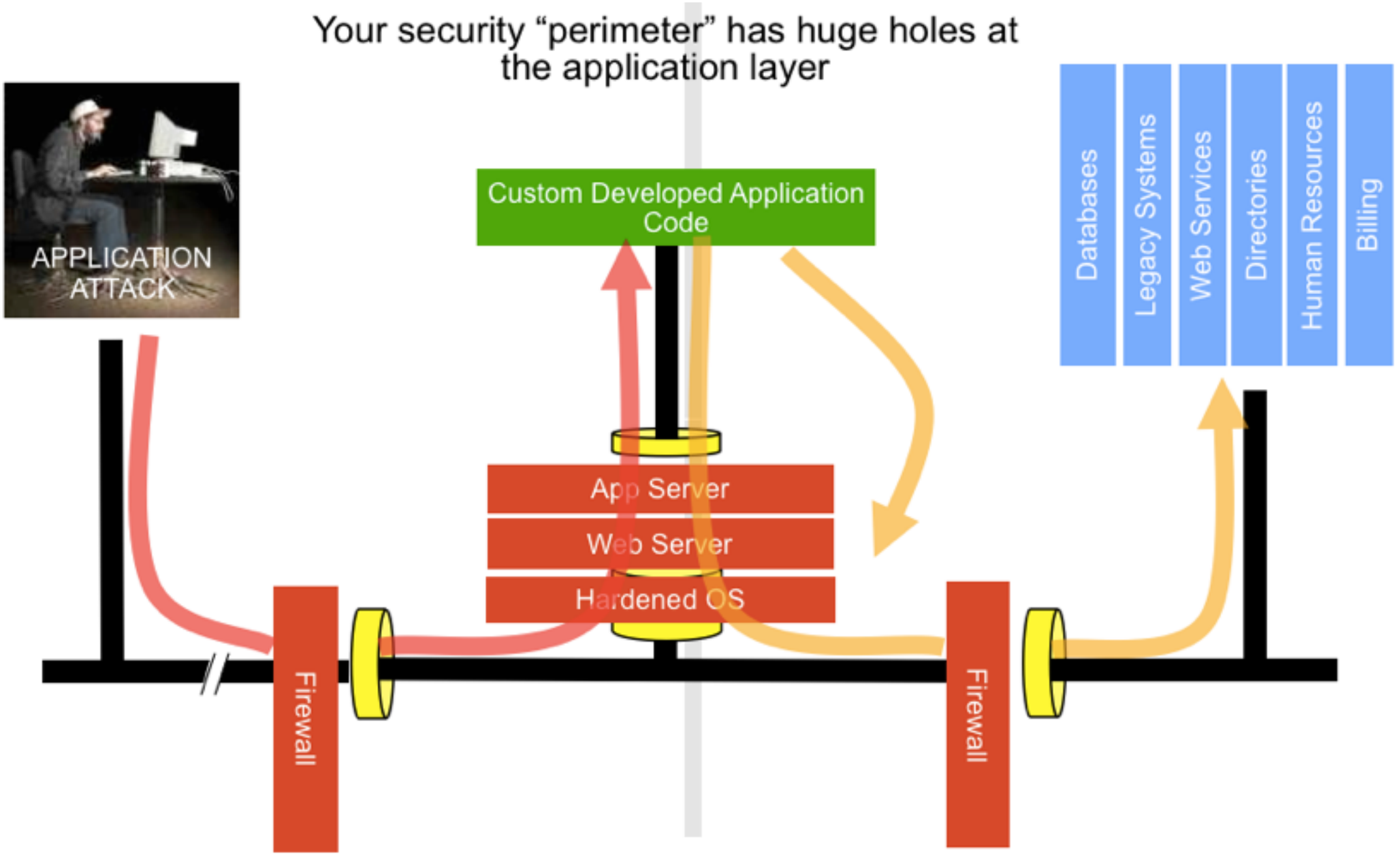
Web Attacking

- Web Defacement
- Malicious script spreading
- Phishing
- Database and Credential stolen

Review: web app security

- Lack of security awareness
- A lot of misunderstanding
  - Network firewall can also protect web applications
  - Security is only network security and ISO standard
- Lack of secure coding skills
- Need web application firewall implemented
- Need web application audit

Why we need web application security?



You can’t use network layer protection (firewall, SSL, IDS, hardening) to stop or detect application layer attacks

Network Security is not enough

- Network Security Mostly Ignores the Contents of HTTP Traffic, such as....
  - Firewalls, SSL, Intrusion Detection Systems
  - Operating System Hardening, Database Hardening
- Need to secure web application (Not Network Security)
  - Securing the "custom code" that drives a web application
  - Securing libraries
  - Securing backend systems
  - Securing web and application servers
- Cloud Computing is coming, the infrastructure is secured by the provider but we are still need to secure our application.

OWASP

- Open Web Application Security Project
- http://www.owasp.org
- Open group focused on understanding and improving the security of web applications and web services!
- Hundreds of volunteer experts from around the world

OWASP
The Open Web Application Security Project
http://www.owasp.org

OWASP

The Open Web Application Security Project

Navigation

- Home
- News
- OWASP Projects
- Downloads
- Local Chapters
- Global Committees
- AppSec Job Board
- AppSec Conferences
- Presentations
- Video
- Press
- Get OWASP Books
- Get OWASP Gear
- Mailing Lists
- About OWASP
- Membership

Reference

- How To...
- Principles
- Threat Agents
- Attacks
- Vulnerabilities
- Controls

Main Page

Welcome to OWASP

the free and open application security community

- OWASP Summit 2011
- Top Ten
- WebScarab
- ESAPI
- ASVS
- AntiSamy

About • Searching • Editing • New Article • OWASP Categories

The Open Web Application Security Project (OWASP) is a 501c3 not-for-profit worldwide charitable organization focused on improving the security of application software. Our mission is to make application security visible, so that people and organizations can make informed decisions about true application security risks. Everyone is free to participate in OWASP and all of our materials are available under a free and open software license.

You'll find everything **about OWASP** here on our wiki and current information

Special

Quick Reference

Election of Officers Upc

Community Forums - C

Contact OWASP Staff -

Industry Citations - Cli

Podcast - Listen Now

Blog - Click Here

EGA e-Government Agency

G-CERT by EGA

OWASP Top 10 2013

- Injection
- Broken Authentication and Session Management
- Cross-Site Scripting(XSS)
- Insecure Direct Object Reference
- Security Misconfiguration
- Sensitive Data Exposure
- Missing Function Level Access Control
- Cross-Site Request Forgery(CSRF)
- Using Components with Known Vulnerability
- Unvalidated Redirects and Forwards

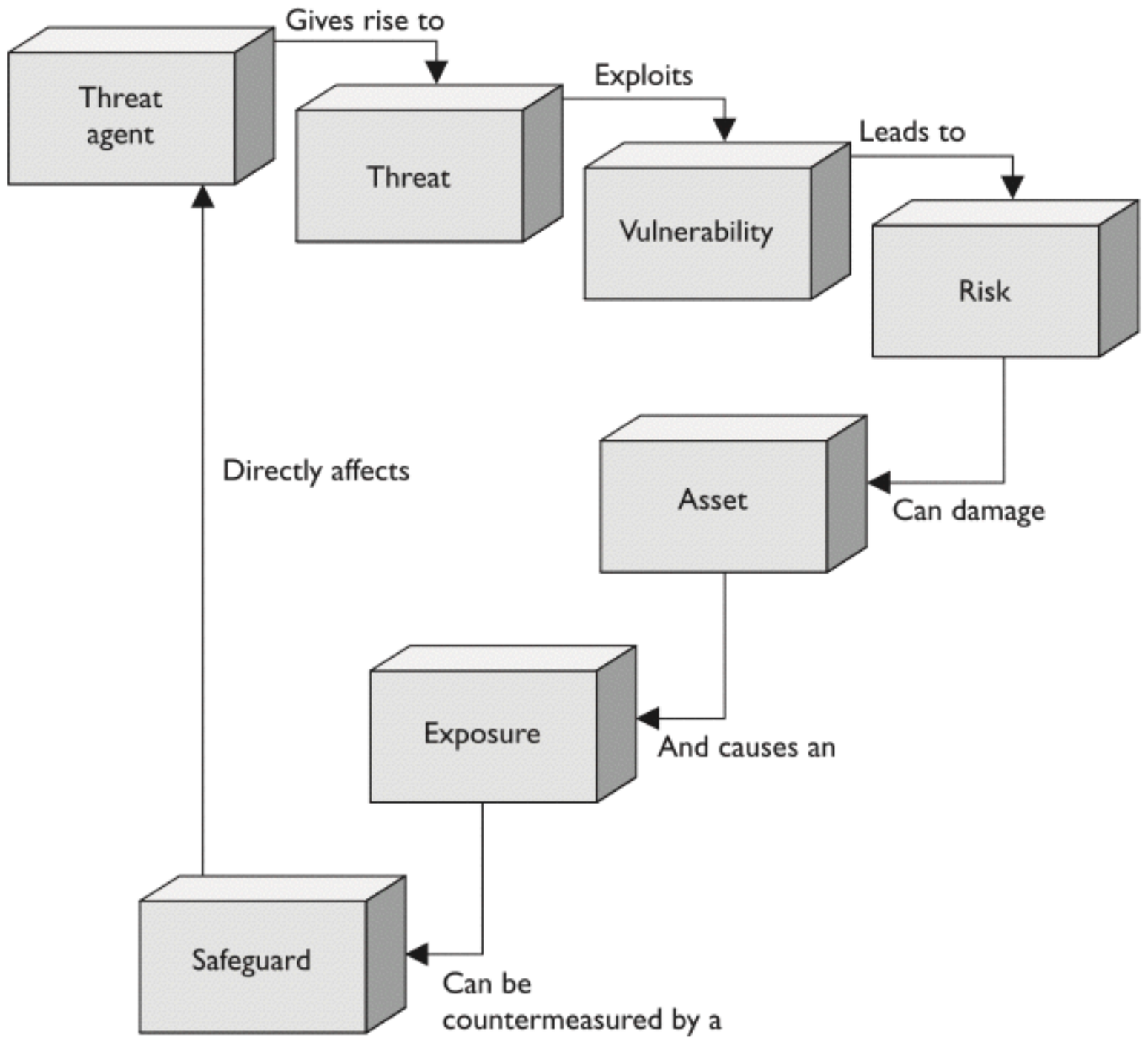
Answer these questions

- How many websites do you have?
- Did you develop by yourself?
- If Yes,
  - Did you use CMS or coding by yourself?
- If No,
  - Who did? Can you control them?
- Is there login page on your website?
- Do you use HTTPS?
- Have you ever updated your servers and apps?

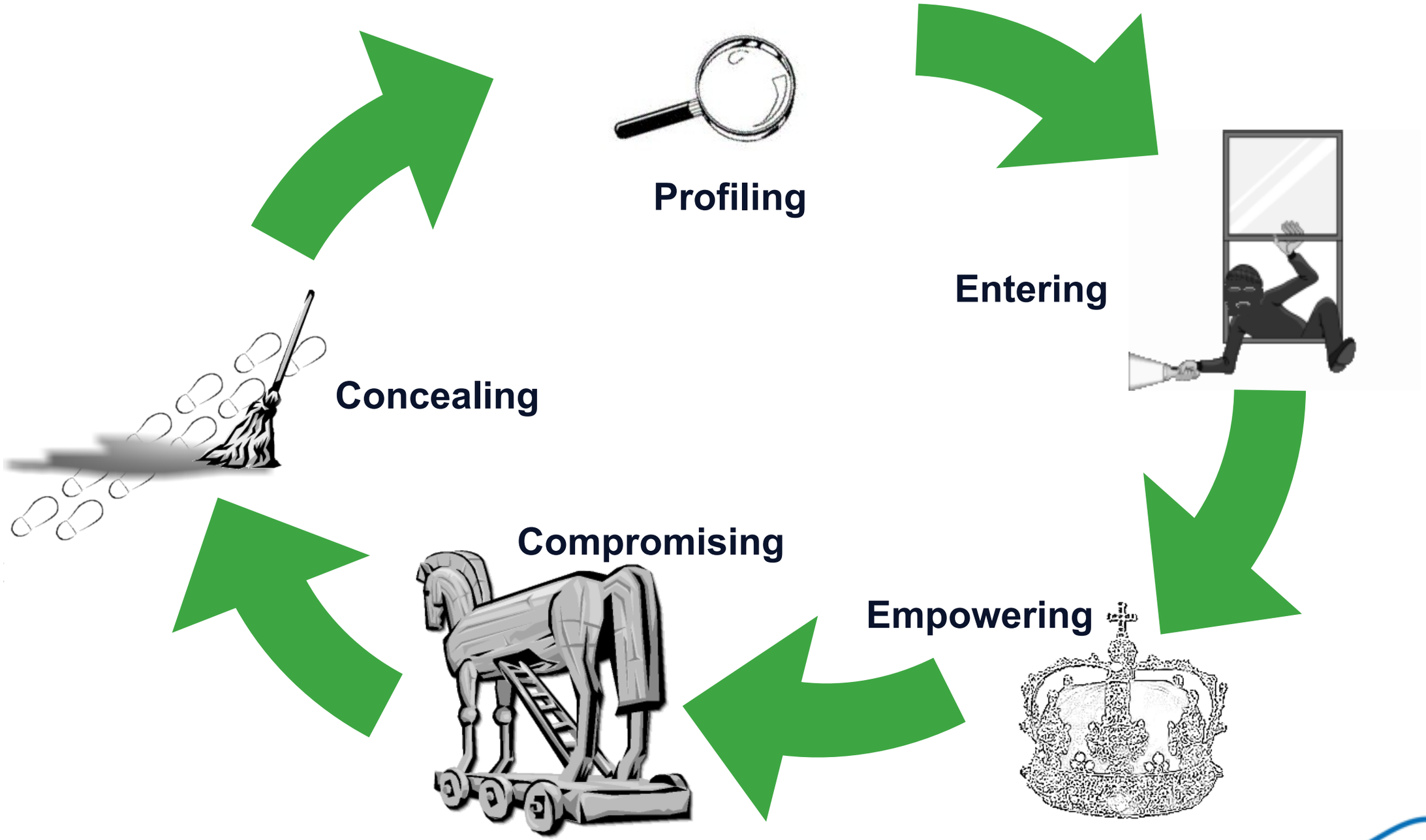
How to attack our servers?

- Systems
  - OS
  - Software installed
- Network
  - Sniffer
  - Spoofing
  - Flooding / DDoS
- Applications
- Data
- Operation

Security components



General Attack Lifecycle



Security by Obscurity

Information Leakage

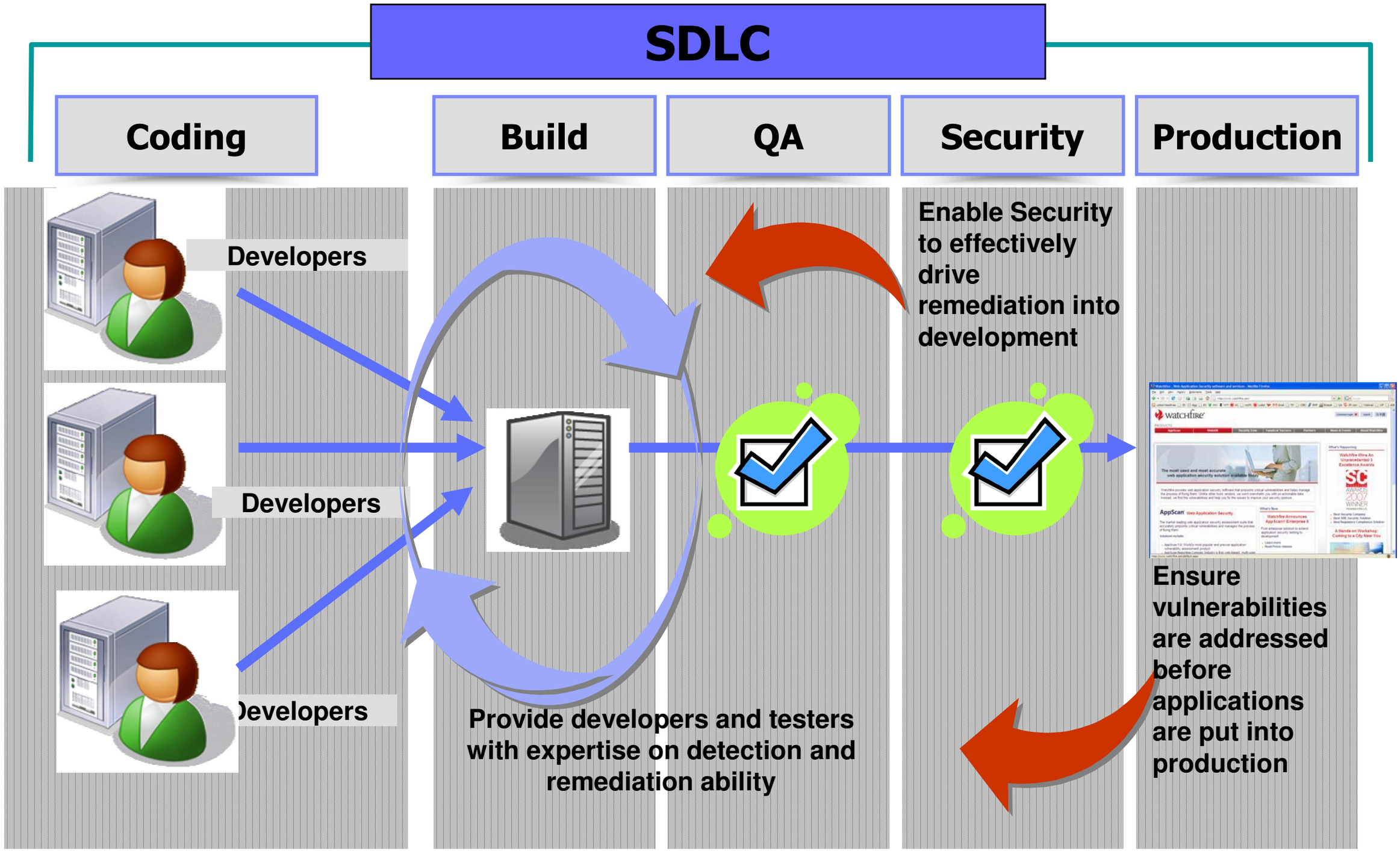
- Remove from OWASP Top 10 vulnerabilities
- Information Leakage
  - Application internals, environment information
  - Reduce the effort to launch a successful attack
  - Results in more targeted attacks
- Security by obscurity
  - Insufficient to properly secure applications
  - Increases the effort for an attacker
  - Increases chances of detecting attack patterns
- General advice: Do not expose information that doesn't need to be exposed

Exercises

- Access your organization’s website
- What information do you disclose unnecessarily?
- How could that information be used in an attack?

Security culture

Building security and compliance into the SDLC



Why people choose not to build secure systems

- Security is boring.
- Security is often seen as a functional disabler.
- Security is difficult to measure.
- Security is usually not the primary skill or interest of designers and developers.
- Security means not doing something exciting and new.
- Plus: Security is often an afterthought, not an integral part of a project.

Source: Michael Howard, David LeBlank, Writing Secure Code, Microsoft Press 2003

Selling security to the organisation

- Secure products are quality products
- If your company doesn't care about creating quality products, find a job elsewhere.

  But: It's not that simple, there is no perfect security.
- The media (and your competition) leap on security issues
- Security vulnerabilities are expensive to fix.

Source: Michael Howard, David LeBlank, Writing Secure Code, Microsoft Press 2003

Security vulnerabilities are expensive to fix

A basic survey of software companies that have established practices for fixing vulnerabilities that lead to attacks approximate that the costs associated with remediating a Web site that has encountered an XSS-like attack is around 40 man-hours per incident.

That cost combined with the cost of hiring or training an engineer to address the problem (~USD 100/hour) and the average number of seven XSS (or similar) exploitable vulnerabilities per Web site brings the total estimated cost to USD 28,000 to fix each problem reactively.

Security breaches can be much more expensive. This figure does not account for the impact to online business transactions, customer satisfaction issues, or other potential risks (compensation for damages, fines) associated with a business' Web site being vulnerable to hijacking, phishing, or defacement.

Source: Microsoft SDL Quick Security Reference - Cross-Site Scripting, http://www.microsoft.com/download/en/details.aspx?id=13759

Raising security awareness : Never waste a crisis

"It seems that all security practitioners struggle with the same predicament: How do I get the software engineering teams to wake up and start taking software security seriously? One of the most effective ways to achieve rapid, dramatic change is to leverage a crisis."

– Brad Arkin, Adobe Systems

Advise for security champions in normal times

- Fight the good fight
  - Be persistent (but not annoying)
  - Build allies in the team
  - Deliver data-driven arguments and appeals for resources
  - Play within the bounds of what process and culture allow
- Build your network throughout the company (Legal, Sales, Marketing, PR, executive management)
- Build your social network
- Have a continually refreshed plan how to respond to crisis
  - Understand the business
  - Plan scenarios based on real security failures
  - Develop a magic-wand plan: “In a world of unlimited resources, we should do X, Y, and Z”
- Develop metrics

Source: Brad Arkin, “Never Waste a Crisis”, IEEE Security and Privacy, vol. 9, no. 3, pp. 82-85, May-June 2011, doi:10.1109/MSP.2011.58

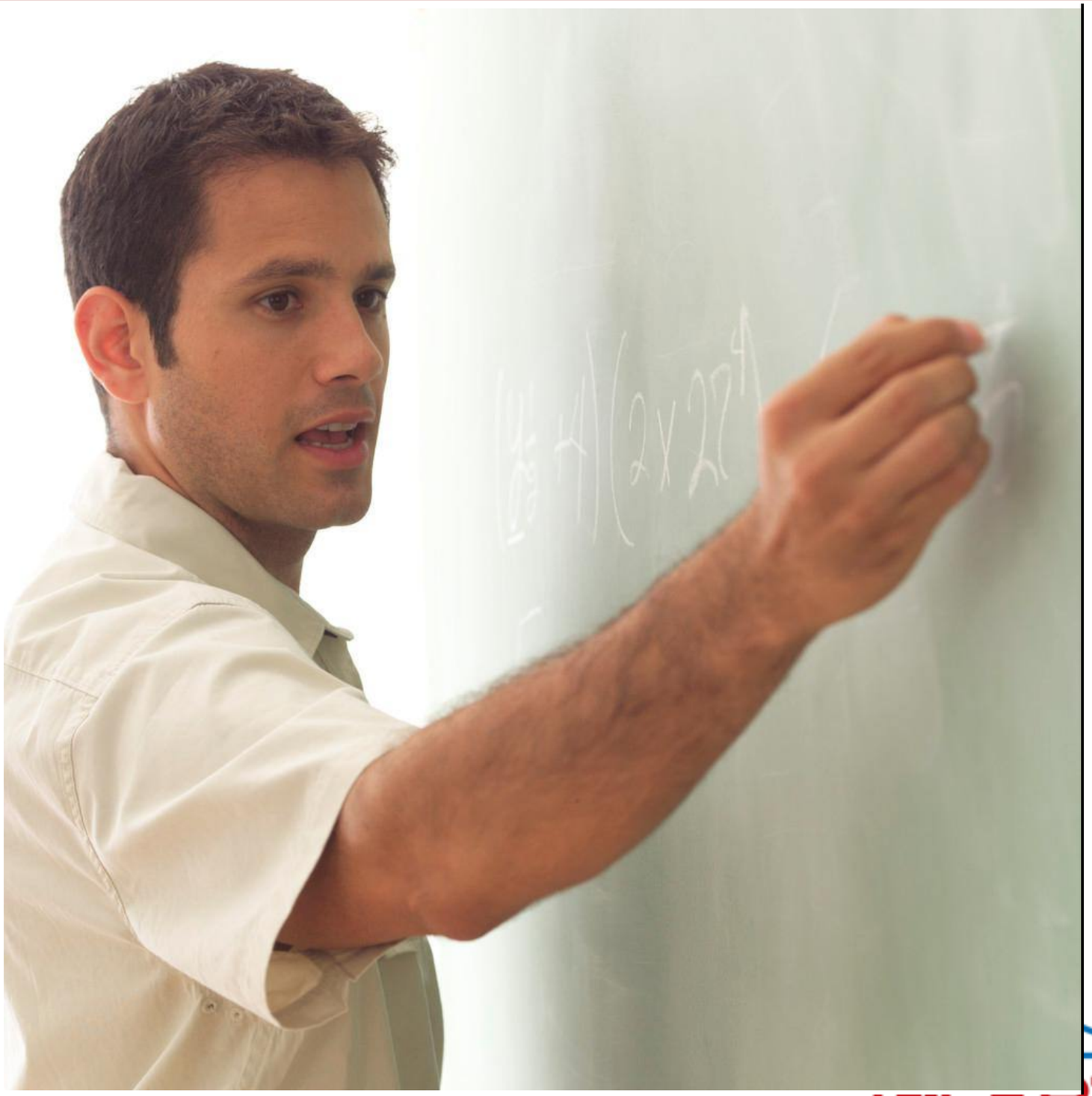
Advise for security champions during a crisis

- Step 1: Speak their language
  - Start with the facts
  - Let go of the detail
  - Convert what you know into the language of your counterparts
  - Provide clear recommendations
- Step 2: Implement the magic-wand plan
  - Link recommendations and crisis
  - Group recommendations in people, process, technology
  - Example: Develop rapid-response capabilities, introduce automated security testing
- Step 3: Be ready to scale
  - Drive long-term culture change
  - Define proven processes, security road map and healthy metrics

Source: Brad Arkin, “Never Waste a Crisis”, IEEE Security and Privacy, vol. 9, no. 3, pp. 82-85, May-June 2011, doi:10.1109/MSP.2011.58

Security education is not for only IT guys

- Foundation for culture change
- IT departments
  - System administrators
  - Architects/designers
  - Developers
  - Testers
- But also
  - Helpdesk staff
  - Business owners
  - Legal department
  - Executive management
- And also
  - Employees
  - Customers
  - Partners
  - Suppliers



General recommendations

- Define application security requirements
  - Consider security aspects at every stage of the project
  - Threat modeling (threat agents, technical and business impact)
  - Include security test cases and ethical hacking
  - Security is not a nice-to-have
- Design applications with a focus on end-to-end security
  - Use security products during development, testing and running applications
  - Use proven libraries and frameworks (and keep them up to date!)
  - Do not rely on network or middleware layers to provide security
  - Think like a hacker (but don't become one :-))
  - "What could go wrong?"
- Define process for dealing with security exposures and incidents
  - Follow security advisories
  - Action plan, e.g. shutdown server, shutdown application
  - Communication plan, e.g. inform affected parties, team leader, executive management

Risk management

Requirements

- People
  - Stakeholders
  - Implementors, managers, admins, data owners and users
- Process
  - Business processes, policies and procedures
  - Input and output
  - Data storage and database
- Technology
  - Network, system and application

Design (1/2)

- Network infrastructure
  - Cloud, data center or hosting
- System used
  - Operating System
  - Softwares or applications
  - Web and Database Servers, Middleware
  - Other tools

Design (2/2)

- Application development
  - CMS or own developing (partial or all)
  - Language
  - Database design
- Data protection
- Operation

Implementation

- Development phase
  - Secure coding
- Production phase (Go live)
  - OS hardening
- Operation
  - Backup and recovery

Testing

- Software testing
- Vulnerability Assessment
- Penetration Testing
- Monitoring
  - Log and system
  - Information

- Audit

Why hackers love your CMS?

- Do you still use Joomla version 1.5 or older?
- Do you use default setting/configuration?
- Do you install unnecessary modules?
- Is your admin's password easy to guess?

- If you answer "Yes", you are vulnerable
- Because exploit code are published (easy to find)
- You can watch the hacking methods on Youtube

If you use CMS, you must .....

- Get Security Announcements
- Choose an active CMS
- Upgrade (Forever)
- Choose plugins/modules wisely
- Back up early and often
- Secure your host
- Use the community

Hardening

- Reduce attack surfaces
- Best suite a newly implemented server before migrating to production
- During production, follow risk assessment reports to add more security
- Both OS and Application

OS Hardening Concept

- Harden Installation : Install package that you want to use.
- Patch and Latest OS
- Backup and Image
- Time Synchronization
- Secure Service : Open Port and Service as you want to serve. Port scan checking
- Network Access Control: Firewall
- Secure User : User Management, Segregate or Duty.

OS Hardening Concept[2]

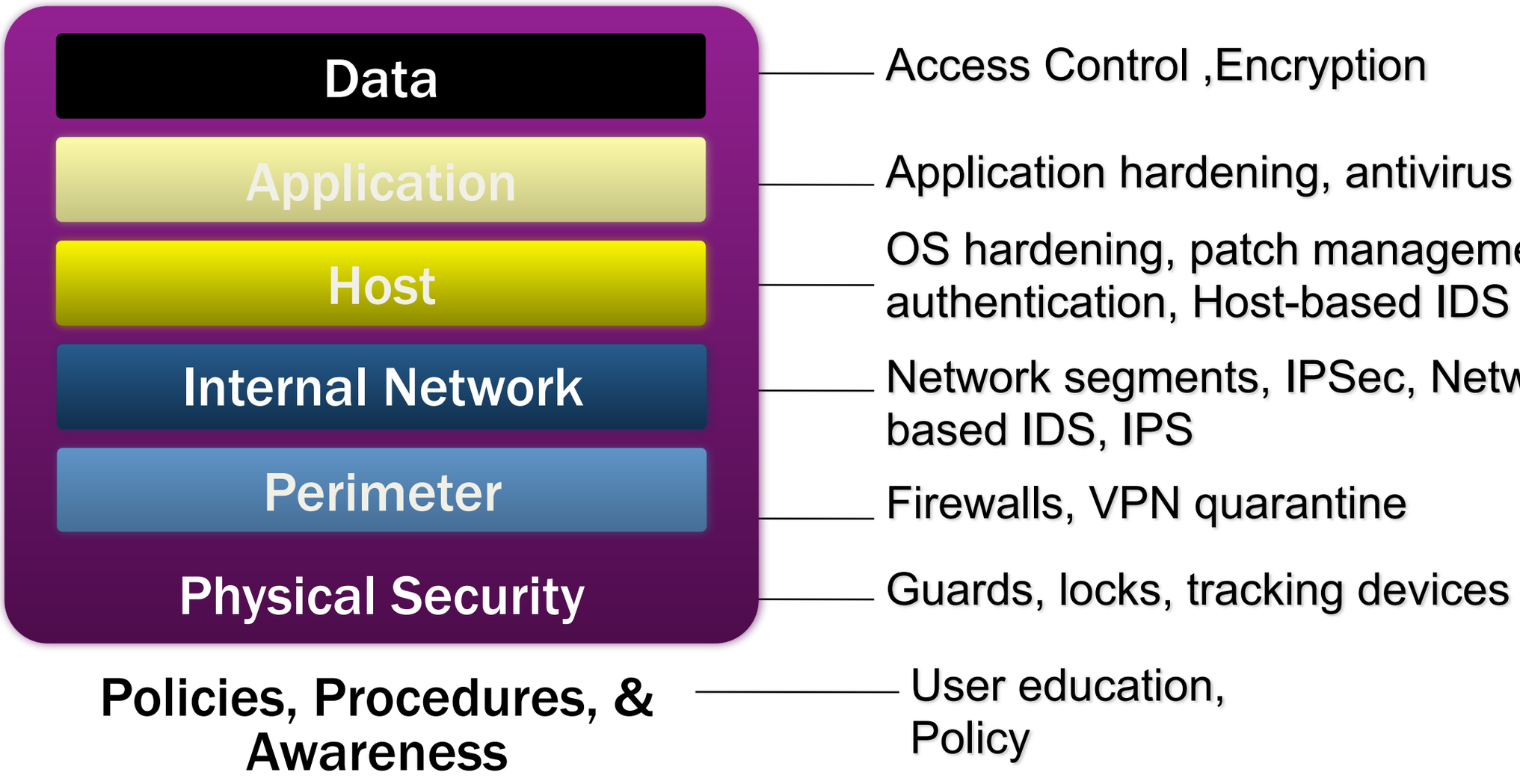
- Secure File and Data : File permission, Owner Control, File integrity checker.
- Harden Kernel
- Secure Administrator communication
- System logging
- System Monitoring
- Antivirus/AntiMalware

Important Main Points

- Update security patches
- Disable unnecessary functions
- Least privilege/access
- Appropriate authentication
- Enable monitoring capabilities
- Secure communications

Conclusion - Defense in Depth

- Using a layered approach:
  - Increases an attacker's risk of detection
  - Reduces an attacker's chance of success



Web Application Security Testing

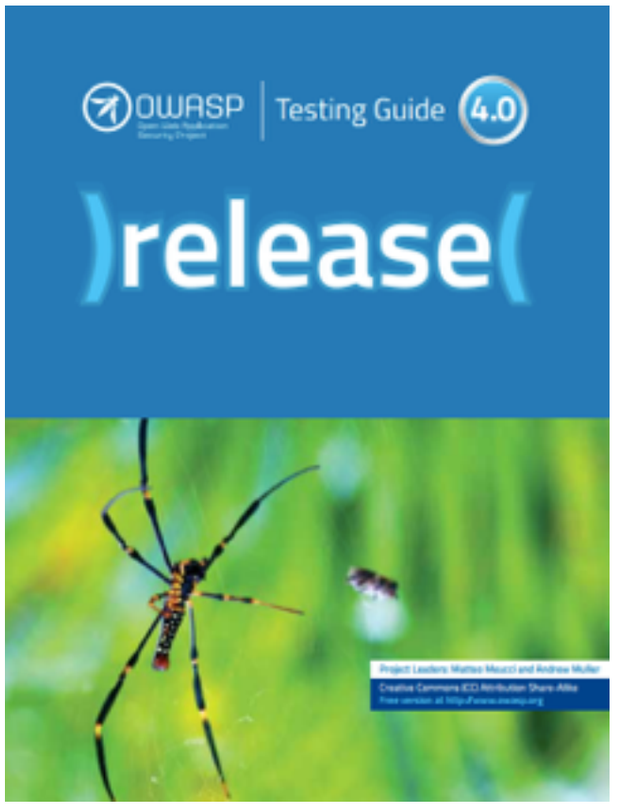
Functional testing

vs

Security testing

OWASP Testing Guide v.4.0

- Most comprehensive open source secure testing guide on the web
- Years of development effort
- Version 4.0 produced 2014
- Hundred of contributors
- Project Leader and Editor
  - Matteo Meucci, Andrew Muller



- www.owasp.org/index.php/Testing_Guide

OTG v4 Index

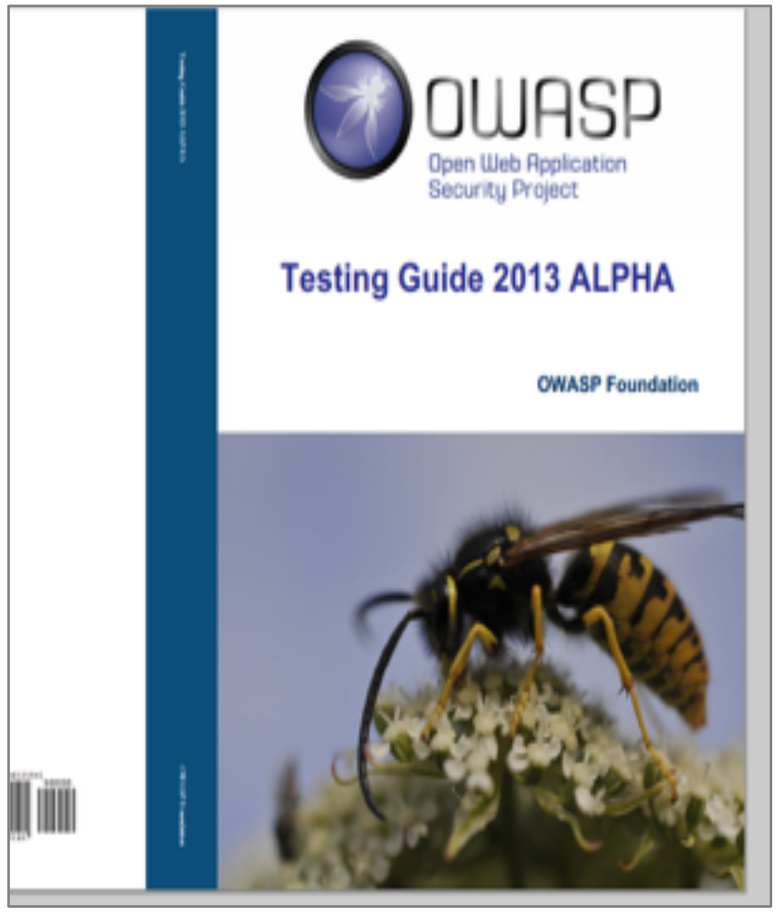
1. Frontispiece
2. Introduction
3. The OWASP Testing Framework
4. Web Application Penetration Testing
5. Writing Reports: value the real risk

Appendix A: Testing Tools

Appendix B: Suggested Reading

Appendix C: Fuzz Vectors

Appendix D: Encoded Injection



Automated Vulner. Scanning tools

- OWASP Zed Attack Proxy (ZAP)
- Burp suite
- w3af
- Acunatix
- Nessus
- .....

An Introduction to ZAP

The OWASP Zed Attack Proxy

The Introduction

- The statement
  - You cannot build secure web applications unless you know how to attack them
- The problem
  - For many developers ‘penetration testing’ is a black art
- The solution
  - Teach basic pentesting techniques to developers

“This was fine for your nephew’s fifth, Sire, but I fear it is set for a sterner test.”

Thanks to Royston Robertson www.roystonrobertson.co.uk for permission to use his cartoon!

The Caveat

- This is in addition to:
  - Teaching secure coding techniques
  - Teaching about common vulnerabilities (e.g. OWASP top 10)
  - Secure Development Software Lifecycle
  - Static source code analysis
  - Code reviews
  - Professional pentesting
  - …

The Zed Attack Proxy

- Released September 2010
- Ease of use a priority
- Comprehensive help pages
- Free, Open source
- Cross platform

- A fork of the well regarded Paros Proxy
- Involvement actively encouraged
- Adopted by OWASP October 2010

More about ZAP

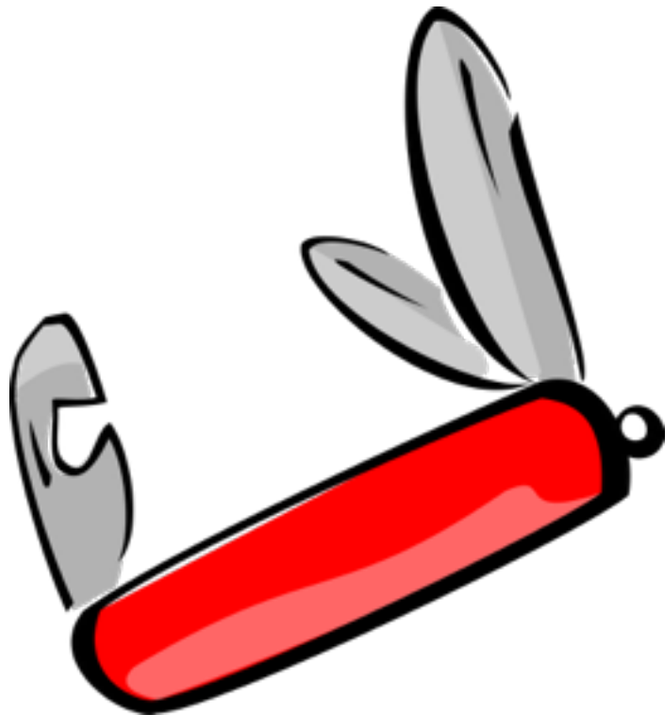
- Project Leader
  - Simon Bennet, UK
- Download from
  - https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

ZAP Principles

- Free, Open source
- Cross platform
- Easy to use
- Easy to install
- Internationalised
- Fully documented
- Involvement actively encouraged
- Reuse well regarded components

The Main Features

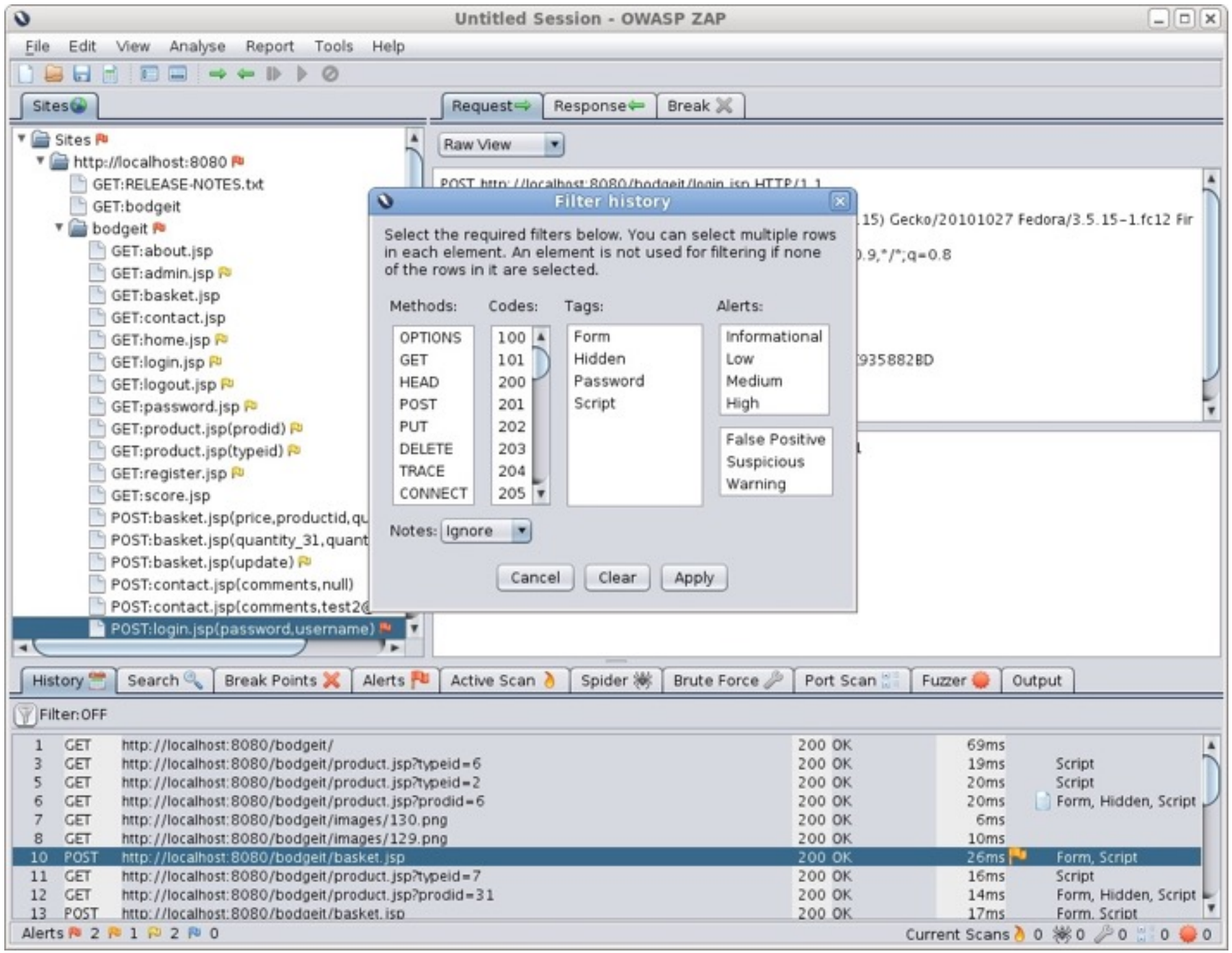
- All the essentials for web application testing
  - Intercepting Proxy
  - Active and Passive Scanners
  - Spider
  - Report Generation
  - Brute Force (using OWASP DirBuster code)
  - Fuzzing (using OWASP JBroFuzz code)



The Additional Features

- Auto tagging
- Port scanner
- Smart card support
- Session comparison
- Invoke external apps
- BeanShell integration
- API + Headless mode
- Dynamic SSL Certificates
- Anti CSRF token handling

The Demo


Untitled Session - OWASP ZAP
File Edit View Analyse Report Tools Help
Sites
Request Response Break
Raw View
Filter history
Select the required filters below. You can select multiple rows in each element. An element is not used for filtering if none of the rows in it are selected.
Methods: OPTIONS GET HEAD POST PUT DELETE TRACE CONNECT
Codes: 100 101 200 201 202 203 204 205
Tags: Form Hidden Password Script
Alerts: Informational Low Medium High
False Positive Suspicious Warning
Notes: Ignore
Cancel Clear Apply
Sites
http://localhost:8080
GET:RELEASE-NOTES.txt
GET:bodgeit
bodgeit
GET:about.jsp
GET:admin.jsp
GET:basket.jsp
GET:contact.jsp
GET:home.jsp
GET:login.jsp
GET:logout.jsp
GET:password.jsp
GET:product.jsp(prodid)
GET:product.jsp(typeid)
GET:register.jsp
GET:score.jsp
POST:basket.jsp(price,productid,qu
POST:basket.jsp(quantity_31,quant
POST:basket.jsp(update)
POST:contact.jsp(comments,null)
POST:contact.jsp(comments,test2@
POST:login.jsp(password,username)
History Search Break Points Alerts Active Scan Spider Brute Force Port Scan Fuzzer Output
Filter:OFF
1 GET http://localhost:8080/bodgeit/ 200 OK 69ms
3 GET http://localhost:8080/bodgeit/product.jsp?typeid=6 200 OK 19ms Script
5 GET http://localhost:8080/bodgeit/product.jsp?typeid=2 200 OK 20ms Script
6 GET http://localhost:8080/bodgeit/product.jsp?prodid=6 200 OK 20ms Form, Hidden, Script
7 GET http://localhost:8080/bodgeit/images/130.png 200 OK 6ms
8 GET http://localhost:8080/bodgeit/images/129.png 200 OK 10ms
10 POST http://localhost:8080/bodgeit/basket.jsp 200 OK 26ms Form, Script
11 GET http://localhost:8080/bodgeit/product.jsp?typeid=7 200 OK 16ms Script
12 GET http://localhost:8080/bodgeit/product.jsp?prodid=31 200 OK 14ms Form, Hidden, Script
13 POST http://localhost:8080/bodgeit/basket.jsp 200 OK 17ms Form, Script
Alerts 2 1 2 0
Current Scans 0 0 0 0 0

Summary and Conclusion 1

- ZAP is:
  - Easy to use (for a web app pentest tool;)
  - Ideal for appsec newcomers
  - Ideal for training courses
  - Being used by Professional Pen Testers
  - Easy to contribute to (and please do!)
  - Improving rapidly

Summary and Conclusion 2

- ZAP has:
  - An active development community
  - An international user base
  - The potential to reach people new to OWASP and appsec, especially developers and functional testers
- ZAP is a key OWASP project

Thank you.